A Modular Multimodal Multi-Object Tracking-by-Detection Approach, with Applications in Outdoor and Indoor Environments

Eduardo Borges[a], Luís Garrote[b] and Urbano J. Nunes[c]
University of Coimbra, Institute of Systems and Robotics, Department of Electrical and Computer Engineering, Portugal
{*eduardo.borges, garrote, urbano*}*@isr.uc.pt*

Keywords: Tracking-by-Detection, AMRs, Point Cloud and RGB, Multimodal and Multi-Object.

Abstract: Object detection and tracking are integral components of numerous modern robotics systems, playing an essential role in applications like autonomous driving and industrial Autonomous Mobile Robots (AMRs). In this paper, we propose a modular multimodal multi-object detection and tracking system tailored for AMRs in complex industrial environments. The proposed system employs a tracking-by-detection approach, utilizing both 3D point cloud and RGB data to detect and track multiple objects simultaneously. To develop it, a baseline unimodal framework was created using a PointPillars detector and the AB3DMOT tracker, operating exclusively on point cloud data. To enhance detection and tracking accuracy, a 2D object detector (YOLOv8) was integrated, enabling multimodal detection. The system's performance was evaluated on the KITTI dataset, demonstrating notable improvements in detection accuracy and tracking consistency. This enhancement strengthens the system's robustness and reliability, which are critical factors for real-time perception in AMRs.

1 INTRODUCTION

As industries embrace the era of automation, catalyzed by the principles of Industry 4.0, the demand for AMRs that can seamlessly navigate intricate and ever-changing environments while avoiding obstacles has intensified greatly. Central to their functionality is the ability to skillfully perceive and interact with the surrounding environment. The avoidance of dynamic objects relies on their continuous monitoring through the detection and tracking of their position, enabling the estimation of their future trajectories. Traditional approaches to object detection and tracking are being eclipsed by the advancements in Deep Learning (DL) techniques. These techniques provide the groundwork for AMRs to operate with unprecedented accuracy, efficiency, and adaptability.

The objective of this work was to develop a real-time system for multi-object detection and tracking, designed for AMRs operating in complex and dynamic industrial environments. The outputs of this system, specifically the object trajectories, will be used to assess the collision risk with objects outside the AMR's security laser field of view. The AMRs will operate in industrial environments performing various tasks depending on their specific types.

An AMR is composed of four main functional components: perception, localization, cognition, and motion control. The perception module is responsible for converting raw sensor data into interpretable information, building an environmental model, and identifying the locations of objects or targets. The system proposed in this paper, designed for real-time multiple object detection and tracking, will be an important part of an AMRs' perception module. Localization uses this data to create local maps and determine the precise position of the AMR within its environment. Cognition, often referred to as the "brain" of the AMR, utilizes the robot's position, local map, external commands, and additional information (*e.g.,* object trajectories) from perception to execute essential functions like path planning and collision avoidance. Finally, motion control handles the navigation decisions from cognition and translates them into commands for the actuators, enabling the AMR to perform its required tasks.

To develop the proposed system, a comprehensive study was conducted on diverse methods of object detection and tracking, with a particular focus on those utilizing Deep Learning techniques. Based

[a] https://orcid.org/0000-0002-4454-6182
[b] https://orcid.org/0000-0003-3833-3794
[c] https://orcid.org/0000-0002-7750-5221

Borges, E., Garrote, L. and Nunes, U.
A Modular Multimodal Multi-Object Tracking-by-Detection Approach, with Applications in Outdoor and Indoor Environments.
DOI: 10.5220/0013073200003822

In *Proceedings of the 21st International Conference on Informatics in Control, Automation and Robotics (ICINCO 2024) - Volume 2*, pages 336-344
ISBN: 978-989-758-717-7; ISSN: 2184-2809

on application-specific criteria, one object detection method and one object tracking method were then selected to build a baseline system capable of meeting the objectives with satisfactory performance. To improve the performance of this baseline system and ensure its effectiveness and robustness, several modifications were implemented and evaluated. Each modification was developed to improve the performance of detection, tracking, or both while minimizing the impact on computational requirements and processing speed, which are critical factors in real-time systems. This work proposes a DL-based multimodal object detection and tracking system for industrial AMRs. To achieve this, a tracking-by-detection approach was employed, where the detection and tracking tasks are performed in sequence. Two systems were developed. The first is an unimodal baseline that uses only point cloud data as input, serving as a reference for comparison. It employs the PointPillars framework (Lang et al., 2019) for 3D object detection and AB3DMOT (Weng et al., 2020a) for Multiple Object Tracking (MOT). The second system expanded upon this baseline by incorporating a 2D object detector, YOLOv8 (Jocher et al., 2023) into the tracking module.

The validation of the proposed approaches was carried out in the KITTI dataset (Geiger et al., 2012). The proposed approaches achieved relevant results, in particular when 2D detections were included in the tracking decision.

The main contributions of this work are:

- A baseline modular system using PointPillars and AB3DMOT for 3D multi-object detection and tracking.
- A multimodal tracking system that integrates 3D point cloud data from LiDAR sensors with 2D image data from RGB cameras using YOLOv8 for 2D object detection.
- Experiments using the KITTI dataset, demonstrating that the integration of 2D RGB detections into the system results in improved performance.

2 RELATED WORK

Multiple Object Tracking (MOT) is an important computer vision task that tracks the trajectories of multiple objects in a sequence of captured data. In the context of autonomous navigation, either indoor or outdoor, it is used as a safety tool to prevent collisions with dynamic entities like people, animals, robots, cars, etc.

MOT approaches can be categorized into two main groups: tracking-by-detection and joint detection and tracking. Tracking-by-detection is a modular approach where the tracking process is decoupled from the detection process. In tracking-by-detection, an object detector localizes the objects in each frame independently and provides them to the tracker. The tracker performs data association and manages trajectories, outputting the *active* trajectories. On the other hand, joint detection and tracking methods perform detection and tracking in a single network. The related work presented will be more focused on tracking-by-detection works, as they are closer to the approach followed in this work. Table 1 summarizes the methods discussed in this section.

Focused on real-time applications, Bewley *et al.* proposed Simple Online and Realtime Tracking (SORT) using the tracking-by-detection approach (Bewley et al., 2016). SORT employs the Faster Region-Based Convolutional Neural Network (R-CNN) (Ren et al., 2016) as a 2D object detector and a Kalman filter (Kalman, 1960) with a constant velocity model to predict the future state of detections. During the data association stage, a cost matrix is calculated by computing the Intersection over Union (IoU) distance between each detection and all predicted bounding boxes, with optimal assignments made using the Hungarian algorithm (Kuhn, 1955). The track management system employed is simple yet effective: unique identities are assigned to objects as they enter or leave the sensor's Field Of View (FOV). Detections not associated with tracks are converted into temporary tracks and monitored until the system gains sufficient confidence to avoid tracking false positives. Tracks were terminated if the system failed to detect them for a specified number of frames.

Wojke *et al.* introduced Deep SORT (Wojke et al., 2017) as an extension of the SORT approach that incorporates appearance information, reducing identity switches and enabling tracking during longer occlusion periods. This information is captured by employing a pre-trained CNN to extract descriptors from each detection. The main distinction between the two methods lies in the computation of the cost matrix. Deep SORT builds this matrix by combining two metrics in a weighted sum: the motion metric and the appearance metric. The motion metric is calculated using the squared Mahalanobis distance (Mahalanobis, 1936) between the predicted Kalman states and the measured states. The appearance metric is calculated as the cosine distance between the stored appearance descriptors from the tracks and the appearance descriptors of new detections.

While SORT and Deep SORT have shown remarkable results in 2D object tracking scenarios, au-

Table 1: Summary of Tracking-by-Detection MOT Methods.

Method	Year	Type	Advantages	Limitations
SORT (Bewley et al., 2016)	2016	2D	Simple and effective. Low computational cost. Real-time tracking.	Limited to 2D tracking. Identity switches due to occlusion.
Deep SORT (Wojke et al., 2017)	2017	2D	Includes appearance information. Reduces identity switches.	Limited to 2D tracking. Higher computational cost.
AB3DMOT (Weng et al., 2020a)	2019	3D	Extends SORT to 3D. Low computational cost. High processing speed.	Only uses motion information.
mmMOT (Zhang et al., 2019)	2019	3D	Jointly learns 2D and 3D features. Robust.	Relies solely on appearance features. Potential feature dominance issues.
GNN3DMOT (Weng et al., 2020b)	2020	3D	Uses both 2D and 3D features. Mitigates feature dominance. Feature information sharing.	Higher computational complexity. Requires training a GNN.

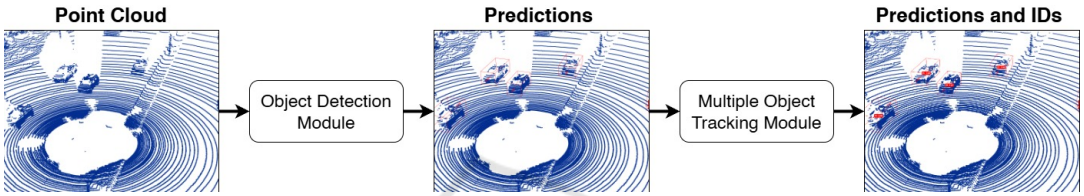


Figure 1: Overview of the object detection and tracking baseline pipeline. The object detector receives LiDAR 3D point cloud data as input to perform object classification and predict 3D bounding boxes. A multiple-object tracker then takes the detector's predictions, assigns an Unique Identifier (UID) to each object, and updates their trajectories over time.

tonomous navigation requires an understanding of the three-dimensional position and movement of objects. Given these considerations, Weng *et al.* proposed A Baseline for 3D Multi-Object Tracking (AB3DMOT) (Weng et al., 2020a), an approach that extends the principles of SORT to the 3D space. Because of this expansion, the 2D detector was replaced by a 3D detector, specifically a pre-trained PointRCNN (Shi et al., 2019). The Kalman filter's state vector was extended to incorporate additional three-dimensional parameters. Consequently, the cost function was adjusted to use the 3D IoU metric. Due to its simplicity and low computational cost, AB3DMOT is one of the fastest methods among 3D MOT systems. The multi-modality Multiple Object Tracking (mmMOT) (Zhang et al., 2019) framework, proposed by Zhang *et al.*, uses a PointPillars detector and solves the association problem using an integer linear programming approach. The cost matrix is obtained by employing a deep adjacency estimation module on 2D and 3D detection features.

The aforementioned methods achieve MOT with varying degrees of success, but some challenges remain. AB3DMOT, similarly to SORT, exclusively uses motion information to build its cost matrix. While this approach might be effective for specific scenarios, including appearance information can enhance discrimination between tracked objects, leading to reduced identity switches and improved accuracy. Deep SORT includes both motion and appearance information, however, it exclusively focuses on 2D or 3D space. Because 2D and 3D information are complementary, using features extracted from both spaces can improve robustness. mmMOT does learn 2D and 3D features jointly, however, it relies solely on appearance-based features and does not address the problem of one type dominating over the other. Furthermore, in all mentioned methods, features are extracted independently of each detection. The association process might improve if a feature from one detected object is informed by the features of each other object. GNN3DMOT (Weng et al., 2020b) was proposed by Weng *et al.* to tackle these challenges. It extracts motion and appearance features from both 2D and 3D spaces, utilizing the Dropout technique (Srivastava et al., 2014) to mitigate feature dominance during training. Additionally, it employs a Graph Neural Network (GNN) to build a feature interaction module that shares feature information between every detection.

3 METHODOLOGY

The system developed in this work will be integrated into a broader system operating on an AMR. This larger system will require object detection for multiple purposes, such as target detection. Therefore,

sharing a single object-detection module would enhance efficiency. With this in mind, a tracking-by-detection approach was adopted for this application. It was also decided that the most suitable next step would be to build the baseline system, using a single detector (unimodal) based on a 3D point cloud sensor, as its main input. Figure 1 provides an overview of the pipeline used to develop the baseline system, highlighting the data flow through its components.

The components of the baseline system were selected from preexisting methods that proved suitable for this use case scenario. The suitability of these methods was evaluated considering their accuracy, efficiency, and simplicity. The accuracy of each method has a direct impact on the system. In this application scenario, accuracy is associated with security and accurate navigation. Poor accuracy can lead to serious injury to people and damage to equipment, resulting in a decrease in trust and adoption of the system. For real-time systems like this, efficiency and speed are crucial to ensure fast decision-making, which is an essential trait to have when navigating dynamic environments.

3.1 Object Detection Module

The purpose of the object detector is to classify and localize the objects of interest in 3D space. Therefore, the search was concentrated on 3D object detectors using point clouds as input data. After some consideration, the chosen 3D object detector was PointPillars (Lang et al., 2019). Considering the KITTI benchmarks, PointPillars ranked among the top performers in terms of accuracy at the time of its publication. Today, its accuracy, while further from the top, still holds up fairly well. Its key strength lies in efficiency, as it avoids the need for computationally expensive 3D convolutions. PointPillars remains one of the fastest methods, achieving a runtime of just 16 ms. Additionally, the accessibility of a publicly available implementation code repository (ZhuLifa, 2022) played a significant role in the decision. This facilitated the implementation and customization to fit this project's requirements and reduced the development time.

3.1.1 Implementation Details

In this work, a reference implementation (ZhuLifa, 2022) inspired by the PointPillars method was used. Despite these differences, this implementation upholds the core principles of PointPillars (Lang et al., 2019). PointPillars is a deep learning model for 3D object detection using point cloud data. It employs a Pillar Feature Net to extract features from the point cloud by converting 3D data into a 2D pseudo-image. These features are then processed through a 2D Convolutional Neural Network (CNN) backbone. Finally, a detection head predicts object bounding boxes, classes, and additional relevant attributes.

The reference implementation contains the PointPillars network, comprising the pillar encoder, the backbone, the neck, and the head. It also includes three main scripts for training, evaluating, and testing the PointPillars network. The training script and supporting functions, such as data augmentation, were not altered, since they were developed to use on the KITTI dataset, which was used to evaluate the final pipeline. This script was utilized to train the PointPillars Network. Likewise, the evaluating script was used without modifications to obtain the network performance metrics. The testing script was developed to display the network bounding box detections for a single frame. Rather than using it directly, the script served as a reference for creating the detection module.

The detection module was implemented as a reusable Python function that can be integrated into a larger application. This function can be called successively to process each frame sequentially, enabling real-time processing. As shown in Figure 1, it receives a LiDAR 3D point cloud, a detection threshold value, and the camera and LiDAR calibration configurations as input. It starts by filtering the point cloud to remove all invalid points (not captured by the camera), thereby improving the inference speed. Then, it runs inference on these filtered points to achieve the predictions. These predictions are finally filtered by confidence, to remove low-confidence predictions, ensuring only the most reliable detections are passed on for tracking. Each detection is represented by:

$$D_{3D} = (f, c, x_1, y_1, x_2, y_2, s, h, w, l, x, y, z, \theta, \alpha)$$

where f denotes the frame number, c the class label, (x_1, y_1) and (x_2, y_2) the coordinates of the top left and bottom right corners of the projected 2D bounding box, respectively. s represents the detection score, (h, w, l) denotes the height, width, and length of the 3D bounding box, and (x, y, z) are the center coordinates of the 3D bounding box, θ is the object's angle of rotation around its Y-axis, and α is the observation angle.

3.2 Multiple Object Tracking Module

The multiple-object tracking module will receive predictions from the PointPillars detector (see Figure 1), in the form of 3D bounding boxes. Therefore, the selected tracker must be capable of handling 3D detections. Among the available methods, meeting this

requirement, AB3DMOT (Weng et al., 2020a) was selected. AB3DMOT is a simple and effective real-time 3D MOT system designed for applications such as autonomous navigation. It focuses on achieving high accuracy while maintaining a low computational cost, making it ideal for real-time applications where high-speed processing is critical. The main factors that contributed to this selection were its simplicity and speed. It employs a 3D Kalman filter for state estimation and the Hungarian algorithm for data association. This straightforward approach results in a lower computational cost and higher processing speed. AB3DMOT is one of the fastest methods on the KITTI tracking benchmark (Geiger et al., 2012). It first obtains current-frame 3D detections from point clouds using an "off-the-shelf" 3D object detector. Next, it employs a 3D Kalman filter to predict the state of associated trajectories to the current frame using a constant velocity model. The Hungarian algorithm is then used to match these predicted trajectories to the obtained 3D detections. Afterward, the 3D Kalman filter updates the state of matched trajectories based on the corresponding matched detections. Additionally, a birth and death memory manages the associated trajectories, adding new ones for newly detected objects and removing those of disappeared objects (Weng et al., 2020a). Similar to PointPillars, its accuracy, when used with a PointRCNN detector (Shi et al., 2019), was among the top performers at the time of publication, continuing to fare well when compared to more recent methods. Furthermore, the availability of the official Python implementation (Weng, 2020) was also a significant factor in the decision, for the same reasons considered in the detector selection.

3.2.1 Implementation Details

The official AB3DMOT repository contains a library with the tracker model and several supporting functions. It also includes a main function to process and visualize object tracking. Initially, this main function receives pre-saved PointRCNN detections, from the KITTI dataset, as input. These detection files are grouped by sequence, sequence type (training or testing), and class. Then, the function executes a loop for every sequence, and inside it a loop for each class, tracking the detected objects. This processing approach, while effective for offline analysis, cannot be employed for real-time applications, as it introduces latency. The developed system needs to handle each frame as it is received to ensure immediate and accurate tracking.

The tracking module was developed in a manner similar to the detection module. It was implemented as a Python function which can be called successively after the detection module to process its detections. The tracking function receives the detections from the current frame along with the frame number. Unlike the previously discussed approach, these detections are not grouped by class, each detection includes a value representing its class. These detections are then organized into a dictionary-like data structure, to ensure compatibility with the tracking model. Afterward, these newly formatted detections are provided to the tracking model to perform tracking and obtain the *active* trajectories T_{3D}. Each trajectory T_{3D} is represented as a modified version of a detection D_{3D}. The tracking module introduces a UID, id, rearranges the order of the existing variables, and omits the frame number f, and observation angle α. The tracking data structure T_{3D} is defined as:

$$T_{3D} = (h, w, l, x, y, z, \theta, c, id, s, x_1, y_1, x_2, y_2) \quad (1)$$

3.3 System Integration

In the integration phase, a main function was developed to simulate real-time object detection and tracking. This behavior was achieved with a loop that reads the current frame input data from memory.[1] This data contains one image, one point cloud, and the current robot pose. The camera and LiDAR calibration parameters are static and do not need to be read. The main function then calls the detection function to process the point cloud and obtain the detections. The predictions from the detection module are then passed to the tracking module to generate the *active* trajectories. This process is repeated for each subsequent frame.

The predictions from the detection function (using the PointPillars network) adhere to the same 3D world coordinate system convention as the KITTI dataset (forward, leftward, upward). However, the AB3DMOT model, used in the tracking module, was developed using PointRCNN, which follows a different coordinate system convention (rightward, downward, forward). These compatibility issues were addressed and both modules were integrated to create the pipeline depicted in Figure 1.

3.4 Integration of a 2D Object Detector

As is, the system uses an adjustable confidence score threshold to categorize detections into two groups: low-confidence detections D^L_{3D}, and high-confidence

[1] In the real system, this data would be delivered directly by the robot's sensors.

detections D_{3D}^{H}. High-confidence detections enter the tracking module, while low-confidence detections are discarded. The tracking module initializes a trajectory only after it has been consistently matched with a detection for a minimum number of consecutive frames min_{hits}. This strategy helps prevent false positives, as matching false detections over successive frames is unlikely. While effective, this strategy can introduce false negatives, as min_{hits} consecutive matches are required to initiate a trajectory, potentially delaying the recognition of true detections. To address this issue, a 2D object detector was integrated into the detection module, running concurrently with the 3D object detector, to independently detect objects of interest in RGB images. The selected 2D object detector was YOLOv8 due to its state-of-the-art accuracy and speed, essential for real-time detection. Additionally, its availability as a Python package made it easy to integrate into the existing system.

Some 3D detections have corresponding 2D detections originating from the same object. Due to their independence, distinct data sources (RGB and point clouds), and differing detector architectures, these detections are more likely to be classified as true positives. As such, they were grouped into a new category: very high confidence detections D_{3D}^{VH}. Detections from this group do not get filtered out and do not need to adhere to the same rules as high-confidence detections. Instead, they are immediately initialized into trajectories by the tracking module if $\max(conf_{2D}, conf_{3D}) > conf\_th$ and $cls_{2D} = cls_{3D}$, where $conf_{2D}$ and $conf_{3D}$ represent the confidence scores from the 2D and 3D detectors, respectively, $conf\_th$ denotes the adjustable confidence threshold, and cls_{2D} and cls_{3D} represent the object class of the 2D and 3D detections, respectively.

This modification aims to improve detection accuracy by reducing false negatives by allowing detection with very high confidence to bypass the min_{hits} requirement, reducing the delay in trajectory initialization. Nevertheless, this requirement is still important for more ambiguous situations, such as those where an object is only detected by the 3D detector (D_{3D}^{H}).

3.4.1 Implementation Details

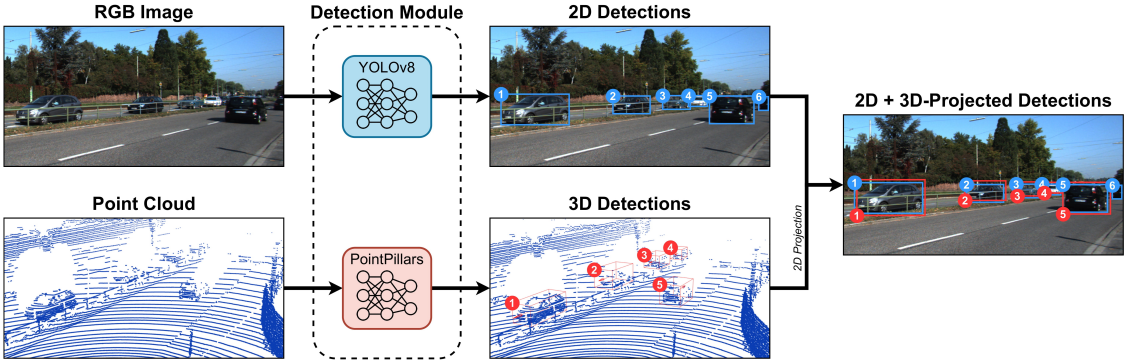
The YOLOv8 detector provides 2D bounding boxes in the format $BB_{2D} = (x_1, y_1, x_2, y_2)$, where (x_1, y_1) represents the coordinates of the top left corner, and (x_2, y_2) represents the coordinates of the bottom right corner of the bounding box. Similarly, the PointPillars detector generates 2D bounding boxes, representing a projection of the predicted 3D bounding boxes to the image plane. These 2D bounding boxes follow the same format: $BB_{2D}^{3D} = (x_1, y_1, x_2, y_2)$. By directly utilizing these bounding boxes, the step of projection that requires camera and LiDAR calibration matrices can be bypassed, thus saving computational resources. Figure 2a provides an overview of the integration process for the 2D and 3D object detectors. Additionally, it illustrates how both types of detections are visualized on the image plane. To identify which 3D detection has a matching 2D detection (D_{3D}^{VH}), a straightforward IoU is calculated for all pairs of bounding boxes (each consisting of one BB_{2D} and one BB_{3D}). The resulting values are organized into a table, commonly referred to as an affinity matrix, representing the overlap between each pair of bounding boxes.

To find the optimal pairing, where each 3D detection is matched with only one 2D detection, the Hungarian algorithm is applied to this table. While the Hungarian algorithm was designed for minimization problems, this task requires maximizing the IoU value to find the most suitable matches. To make it compatible with the algorithm, the IoU values from the table are negated, transforming the problem into a minimization task. Low-overlap matches are then filtered out by applying an IoU threshold to the resulting pairs. This process is illustrated in Figure 2b.

4 EXPERIMENTAL VALIDATION

The KITTI dataset (Geiger et al., 2012) was selected for the system evaluation. KITTI is well-established and has become a standard benchmark for evaluating the performance of object detection and tracking algorithms. Furthermore, due to its extensive use in research, it allows for the comparison of the system's performance against a range of existing methods.

The Higher Order Tracking Accuracy (HOTA) benchmark (Luiten et al., 2020) was chosen as the evaluation metric for its ability to address limitations of previous metrics like Multiple Object Tracking Accuracy (MOTA). HOTA offers a comprehensive assessment by evaluating detection, association, and localization through a family of submetrics, ensuring a balanced evaluation across multiple localization thresholds. It also includes Localisation Accuracy (LocA), measuring how accurately predicted bounding boxes match the ground truth, enhancing object localization assessment. The decomposition into components like Detection Accuracy (DetA), Association Accuracy (AssA), Detection Recall (DetRe), Detection Precision (DetPr), Association Recall (AssRe), and Association Precision (AssPr) provides detailed insights into the system's tracking performance.



(a) Overview of the integration of 2D and 3D object detectors into the detection module and visualization of both types of detections on the image plane.

IoU Table	D^1_{2d}	D^2_{2d}	D^3_{2d}	D^4_{2d}	D^5_{2d}	D^6_{2d}
D^1_{3d}	**0.90**	0.00	0.00	0.00	0.00	0.00
D^2_{3d}	0.00	**0.85**	0.00	0.00	0.00	0.00
D^3_{3d}	0.00	0.00	**0.70**	0.00	0.00	0.00
D^4_{3d}	0.00	0.00	0.05	**0.80**	0.00	0.00
D^5_{3d}	0.00	0.00	0.00	0.00	**0.75**	0.00

Hungarian Algorithm → Low-overlap Matches Filtering →

Matches	
D^1_{2d}	D^1_{3d}
D^2_{2d}	D^2_{3d}
D^3_{2d}	D^3_{3d}
D^4_{2d}	D^4_{3d}
D^5_{2d}	D^5_{3d}

(b) Illustration of the IoU-based data association process. The matched detections belong to the very high confidence detection class (D^{VH}_{3D}).

Figure 2: Integration of the YOLOv8 object detector to create a new class of very high confidence detection D^{VH}_{3D}.

4.1 Evaluation Protocol

The systems were evaluated by systematically altering their parameters to identify the combination that produces the best performance. The selection strategy involved adjusting each parameter individually while keeping the others constant to avoid influencing the outcome. After evaluating the performance for each adjustment, the parameter value that produced the best performance was fixed. This performance was determined by averaging the results between all classes. This procedure was repeated for all parameters until the best-performing values for each were identified. The evaluation was conducted using the TrackEval code repository (Luiten, 2020), starting with the determination of the best parameters for the baseline system. Once selected, these parameters were fixed, and the modification was applied individually to create a modified version of the baseline system.

4.2 Baseline System

The baseline system contains three tunable parameters: *m_th*, which sets the affinity threshold for a valid match during data association; *min_hits*, which defines the minimum number of matched detections required for a trajectory to transition from *tentative* to *active*; and *max_age*, which determines the maximum number of consecutive frames a trajectory can remain unmatched before being terminated.

The results were generated using the training sequences, as Ground Truth (GT) information for the test sequences is not publicly available. Although this evaluation approach does not guarantee performance on the test set, the results remain valuable for assessing performance changes resulting from system modifications. Table 2 presents the obtained performance metrics for the “Car” and “Pedestrian” classes using the HOTA benchmark.

From the analysis of the results presented in Table 2, the parameter combination yielding the highest HOTA score for the ”Car” class is BASE_07, while for the ”Pedestrian” class, is BASE_08. However, when considering the cumulative score, the BASE_02 parameter combination produced the best results (with an average of 55.33%). As such, these parameters were selected as the baseline parameters.

4.3 Multimodal Tracking by Detection Pipeline

With the addition of a 2D object detector, the resulting system contains two additional tunable parameters: *iou_th*, which defines the threshold for matching 2D

Table 2: Baseline system evaluation parameters and results for the "Car" and "Pedestrian" classes in the HOTA benchmark.

m_th	min_hits	max_age	HOTA	DetA	AssA	DetRe	DetPr	AssRe	AssPr	LocA
					Car					
0	3	4	68.12%	66.11%	70.41%	73.42%	80.81%	76.29%	85.26%	87.64%
-0.2	3	4	68.22%	66.17%	70.54%	73.53%	80.76%	76.43%	85.26%	87.63%
-0.4	3	4	68.11%	66.15%	70.35%	73.61%	80.62%	76.42%	84.95%	87.33%
-0.2	4	4	67.33%	65.02%	69.94%	71.83%	81.21%	75.53%	85.44%	87.75%
-0.2	5	4	66.31%	63.70%	69.23%	70.06%	81.51%	74.55%	85.55%	87.47%
-0.2	6	4	65.18%	62.30%	68.29%	68.27%	81.72%	73.44%	85.67%	**87.94%**
-0.2	**3**	3	**68.89%**	**67.19%**	**70.86%**	72.97%	**83.03%**	75.48%	**87.03%**	87.72%
-0.2	**3**	5	67.42%	64.99%	70.15%	**73.90%**	78.57%	**77.21%**	83.69%	87.58%
					Pedestrian					
0	3	4	42.33%	42.05%	42.80%	50.22%	56.06%	46.51%	66.47%	72.97%
-0.2	3	4	42.44%	**42.12%**	42.96%	50.36%	56.00%	47.01%	65.29%	72.96%
-0.4	3	4	42.43%	42.08%	42.98%	50.34%	55.95%	47.16%	64.89%	72.95%
-0.2	4	4	42.38%	42.04%	42.93%	49.10%	57.56%	46.91%	65.29%	73.03%
-0.2	5	4	42.14%	41.68%	42.80%	47.92%	58.62%	46.68%	65.52%	73.08%
-0.2	6	4	41.81%	41.19%	42.63%	46.73%	**59.52%**	46.45%	65.62%	**73.12%**
-0.2	**3**	3	40.93%	41.83%	40.25%	48.49%	58.03%	43.43%	**67.19%**	73.05%
-0.2	**3**	5	**43.20%**	41.75%	**44.88%**	**51.19%**	54.37%	**49.60%**	64.72%	72.93%

bounding boxes from the 2D and 3D detectors, and $conf\_th$, which sets the minimum confidence score required for very high confidence detections, D_{3D}^{VH}, to be directly initialized into an active trajectory by the tracking module. Table 3 presents the parameter combinations used to evaluate the system and the obtained performance metrics for the "Car" and "Pedestrian" classes using the HOTA benchmark.

An analysis of the results presented in Table 3 shows that all tested parameter combinations resulted in a significant boost in performance. This modification increased HOTA scores by at least 1.85 and 1.92 percentage points for the "Car" and "Pedestrian" classes, respectively. The best-performing parameter combination (MOD1_03) produced improvements of 2.64 and 3.00 percentage points for the respective classes.

For the "Car" class, the only components of the HOTA metric that showed a decrease in performance were DetPr, AssPr, and LocA. LocA experienced a very slight decrease (from 87.63% to 87.14%), while DetPr and AssPr saw more significant reductions (DetPr dropping from 80.76% to 79.14%, and AssPr from 76.43% to 74.84%). Despite these decreases, both DetA and AssA showed notable improvements, with DetA increasing from 66.17% to 69.67%, and AssA from 70.54% to 72.85%. For the "Pedestrian" class, none of the HOTA components showed a significant decrease in performance. Similarly to the "Car" class, DetA and AssA both increased, with DetA rising from 42.12% to 45.89%, and AssA from 42.96% to 45.23%, reflecting a general improvement in the tracking and detection accuracy. This analysis suggests that while some metric scores slightly declined, the system overall benefited from better detection and association accuracy.

5 CONCLUSIONS

In this paper, we propose a modular multimodal multi-object detection and tracking system designed for operating in indoor or outdoor environments. The system integrates both 3D point cloud data and RGB images, utilizing a tracking-by-detection approach with PointPillars for 3D detection and YOLOv8 for 2D detection. Our experimental results, validated on the KITTI dataset, showed significant improvements in detection accuracy and tracking consistency, particularly due to the integration of 2D and 3D detections, which enhanced robustness. The proposed system showed improvements in both the tracking of vehicles and pedestrians, offering a more reliable solution for real-time perception in dynamic environments.

Several routes for future work can be explored. First, extending the system to incorporate additional sensor modalities, such as thermal or RGB-D data, could further improve detection accuracy in challenging indoor conditions, such as poor lighting. Second, optimizing the computational efficiency of the system for deployment could be another necessary improvement. Third, additional refinement of the object association process, particularly for occluded objects, may help reduce identity switches and improve tracking performance. Lastly, a key direction for future work involves adapting the system specifically for AMRs in industrial environments.

Table 3: Results for the "Car" and "Pedestrian" classes with the multimodal pipeline, considering the 2D detector.

ID	iou_th	conf_th	HOTA	DetA	AssA	DetRe	DetPr	AssRe	AssPr	LocA
Car										
BASE_02	–	–	68.22%	66.17%	70.54%	73.53%	80.76%	76.43%	85.26%	87.63%
MOD1_01	0.5	0.5	70.66%	69.32%	72.28%	78.21%	79.63%	78.99%	84.63%	87.23%
MOD1_02	0.75	0.5	70.07%	68.62%	71.78%	76.85%	**80.27%**	78.17%	**84.95%**	**87.45%**
MOD1_03	0.5	0.3	**70.86%**	69.65%	**72.35%**	79.04%	79.16%	**79.30%**	84.81%	87.14%
MOD1_04	0.5	0.1	**70.86%**	**69.67%**	72.34%	**79.08%**	79.14%	**79.30%**	84.41%	87.13%
Pedestrian										
BASE_02	–	–	42.44%	42.12%	42.96%	50.36%	56.00%	47.01%	65.29%	**73.09%**
MOD1_01	0.5	0.5	44.36%	44.64%	44.30%	54.10%	56.03%	48.66%	65.36%	73.08%
MOD1_02	0.75	0.5	42.96%	42.98%	43.12%	51.69%	**56.08%**	47.25%	**65.43%**	**73.09%**
MOD1_03	0.5	0.3	**45.44%**	45.86%	**45.23%**	56.17%	55.72%	**50.00%**	65.07%	73.07%
MOD1_04	0.5	0.1	45.41%	**45.89%**	45.14%	**56.27%**	55.65%	49.91%	64.99%	73.06%

ACKNOWLEDGMENTS

This work has been supported by the Portuguese Foundation for Science and Technology (FCT) through grant UIDB/00048/2020 and by Agenda "GreenAuto: Green innovation for the Automotive Industry", with reference PRR-C644867037-00000013.

REFERENCES

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jocher, G., Chaurasia, A., and Qiu, J. (2023). Ultralytics yolov8. https://github.com/ultralytics/ultralytics. Accessed: 2024-06-5.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. (2019). Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705.

Luiten, J. (2020). Trackeval: Hota (and other) evaluation metrics for multi-object tracking (mot). https://github.com/JonathonLuiten/TrackEval. Accessed: 2024-05-02.

Luiten, J., Osep, A., Dendorfer, P., Torr, P., Geiger, A., Leal-Taixe, L., and Leibe, B. (2020). Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, 129(2):548–578.

Mahalanobis, P. C. (1936). On the generalized distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49–55.

Ren, S., He, K., Girshick, R., and Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149.

Shi, S., Wang, X., and Li, H. (2019). Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Weng, X. (2020). Ab3dmot: 3d multi-object tracking: A baseline and new evaluation metrics. https://github.com/xinshuoweng/AB3DMOT. Accessed: 2024-02-20.

Weng, X., Wang, J., Held, D., and Kitani, K. (2020a). 3d multi-object tracking: A baseline and new evaluation metrics. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Weng, X., Wang, Y., Man, Y., and Kitani, K. M. (2020b). Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6499–6508.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*. IEEE.

Zhang, W., Zhou, H., Sun, S., Wang, Z., Shi, J., and Loy, C. C. (2019). Robust multi-modality multi-object tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2365–2374.

ZhuLifa (2022). Pointpillars: Fast encoders for object detection from point clouds. https://github.com/zhulf0804/PointPillars. Accessed: 2024-02-20.